BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 13/28	A1	(11) International Publication Number: WO 99/63447
		(43) International Publication Date: 9 December 1999 (09.12.99)

(21) International Application Number: PCT/US99/04610

(22) International Filing Date: 1 March 1999 (01.03.99)

(30) Priority Data:
09/088,133 1 June 1998 (01.06.98) US

(71) Applicant: ADVANCED MICRO DEVICES, INC. [US/US]; One AMD Place, Mail Stop 68, Sunnyvale, CA 94088–3453 (US).

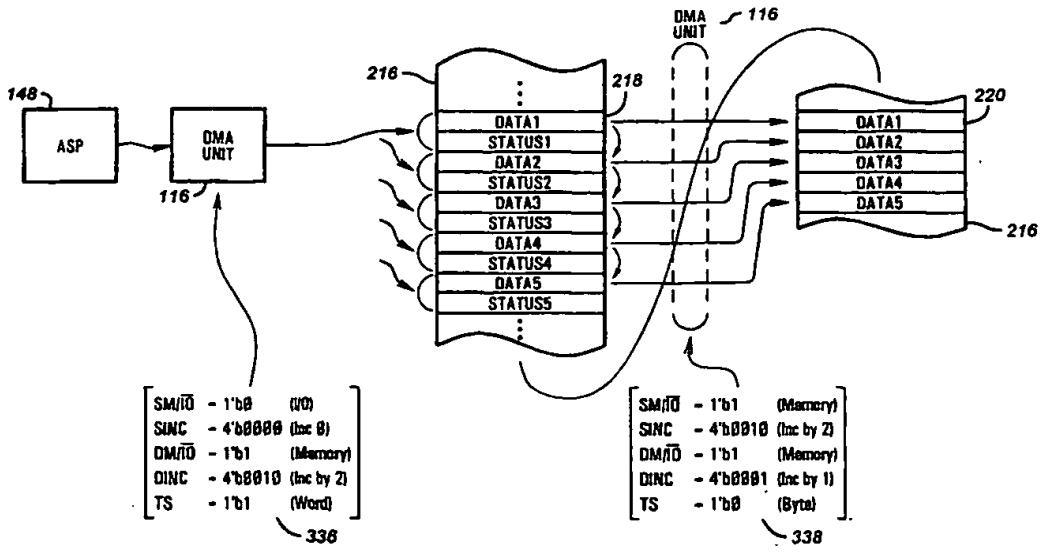
(72) Inventors: SPILO, David, A.; 4200 Balcones Woods Drive, Austin, TX 78759 (US). TYPALDOS, Melanie, D.; 700 Jerry's Lane, Buda, TX 78610 (US).

(74) Agent: APPERLEY, Elizabeth, A.; Advanced Micro Devices, Inc., 5204 East Ben White Boulevard, Mail Stop 562, Austin, TX 78741 (US).

(81) Designated States: JP, KR, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: COMPRESSION AND DECOMPRESSION OF SERIAL PORT DATA AND STATUS USING DIRECT MEMORY ACCESS



(57) Abstract

A microcontroller (M) includes a direct memory access unit (116) that compresses and decompresses data and transfers from one block of memory to another. Specifically, word size data can be read, one byte discarded, and stored as consecutive, byte size data. This can be used in conjunction with an extended read and extended write asynchronous serial port (148) that stores status information along with data. Once the status information is processed, the status is stripped by performing the "compressive" DMA.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

APPLICATION FOR PATENT

TITLE: COMPRESSION AND DECOMPRESSION OF SERIAL PORT DATA AND STATUS USING DIRECT MEMORY ACCESS

SPECIFICATION

TECHNICAL FIELD

1. Field of the Invention

The invention relates to microcontrollers, and more specifically, to a microcontroller providing an improved direct memory access controller for use in conjunction with a serial interface.

BACKGROUND ART

Specialized microcontrollers with integrated communication features are becoming particularly attractive for communications applications. A microcontroller, or an embedded controller, is uniquely suited to combining functionality onto one monolithic semiconductor substrate (i.e. chip). By embedding various communication features within a single chip, a communications microcontroller may support a wide range of communication applications.

Microcontrollers have been used for many years in many applications. A number of these applications involve communications over electronic networks, such as telephone lines, computer networks, and local and wide area networks, in both digital and analog formats. In communications applications, a microcontroller generally has a number of integrated communications peripherals in addition to the execution unit. These can be low and high speed serial ports, as well as more sophisticated communications peripherals, such as a universal serial bus (USB) interface, and high level data link control (HDLC) channels.

An asynchronous serial communications port is one such common additional feature in a microcontroller. An asynchronous serial link allows the microcontroller to communicate with other devices or over data lines by sequentially sending and receiving bits of data. The "asynchronous" nature indicates these ports do not provide a separate clock signal to clock the data. Instead, the rate at which data is sent and received must be predetermined or prenegotiated, and independently controlled on both the sending and receiving ends. This data rate is known as the baud rate, which is the inverse of one bit period. The baud rate is generally one of a number of predefined rates, which are standard within the industry. Such rates include 1200, 2400, 4800, 9600, 19.2K, 28.8K, 33.3K, and 54K baud and high data transfer rates.

Due to the prevalence of serial data communication, many microcontrollers include one or more asynchronous serial ports (ASPs) which can transmit and receive data one bit at a time. Such microcontrollers typically employ interrupt signals to notify the microprocessor core that an ASP requires services. An ASP typically issues an interrupt request signal when a data unit has been received by the ASP and needs to be transferred from the ASP to an external memory unit, when the ASP is finished transmitting a data unit and the next data unit to be transmitted must be transferred from the external memory unit to the ASP, or when an error occurs.

An ASP can be configured for a variety data formats, although historically seven or eight data bits are typical values. A number of nine-bit serial protocols, however, have been developed using microcontrollers, including a nine-bit asynchronous serial protocol in conjunction with direct memory access. Such protocols are described in U.S. patent application Serial No. _____, entitled A MICROCONTROLLER WHICH IS CONFIGURABLE TO TRANSFER DATA TO AND FROM ONE OR MORE ASYNCHRONOUS SERIAL PORTS USING DIRECT MEMORY ACCESS, filed February 4, 1997, by John P. Hansen and Melanie D. Typaldos, and U.S. patent application Serial No._____, entitled A MICROCONTROLLER HAVING HARDWARE FEATURES SUPPORTING 9-BIT SERIAL PROTOCOLS DURING DMA DATA TRANSFERS TO AND FROM ONE OR MORE ASYNCHRONOUS SERIAL PORTS, filed February 4, 1997, by John P. Hansen, Ronald W. Stents, and Melanie D. Typaldos, both of which are commonly assigned and hereby incorporated by reference. These protocols are also described in the Am186ES Users Manual and Am186ED Users Manual, both by Advanced Micro Devices, Inc. of Sunnyvale, California. As described in those applications, and as discussed below, a separate control bit is set or reset to act as the ninth data bit during transmission and reception of data. To support DMA using such 9-bit protocols, when that particular bit is received as a certain value, an interrupt is caused to indicate that the ninth data bit has in fact been set.

DISCLOSURE OF THE INVENTION

According to the invention, a microcontroller includes a direct memory access (DMA) controller that provides compression and decompression of data. Specifically, the DMA controller has source and destination increment values that can be adjusted and are independent of the transfer item size. A source pointer and a destination pointer can be set to blocks in memory, with the source block containing word sized data whose top byte is to be discarded. The source can be set to increment by two after each transfer, and the destination by one. Then, DMA is performed transferring a byte of data from each word in the source, incrementing to the next word, and incrementing to the next byte for the destination. In this way, one byte of data is stripped from each word using DMA with little processor intervention.

This is especially useful when employing an asynchronous serial port that supports extended reads in conjunction with DMA. Sequences of data can be read from the serial port, with each 7 or 8 bit value having a corresponding status byte stored by the DMA controller. Once these status bytes are examined for address bits or error bits, the DMA can be programmed to "strip" the unneeded byte of data.

Similarly, before performing a DMA transfer, byte level data can be expanded into words, with extended write bytes appended to the front of data for transfer by the DMA controller to the asynchronous serial ports supporting extended writes.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1A is a block diagram of a typical microcontroller implemented according to the present invention;

Figure 1B is a schematic pinout diagram of the pinouts for the microcontroller of Figure 1A;

Figure 2 is a block diagram illustrating the relationship between the asynchronous serial port, the DMA controller, and their registers to the microcontroller of Figures 1A and 1B;

Figure 3 is a timing diagram illustrating the use of address bits within an asynchronous serial frame according to the invention;

Figure 4 is a diagram illustrating the DMA unit data compression according to the invention;

Figure 5 is a diagram illustrating the DMA unit data expansion according to the invention;

Figures 6A-6E are block diagrams illustrating register contents in a DMA unit according to the invention;

Figures 7A-7G are block diagrams illustrating the register contents in the asynchronous serial port according to the invention; and

Figure 8 is a block diagram illustrating the implementation of multidrop asynchronous protocol.

MODE(S) FOR CARRYING OUT THE INVENTION

RELATED APPLICATION

The following related applications are hereby incorporated by reference:

U.S. Patent Application Serial No. 08/920,930 entitled A METHOD AND APPARATUS FOR SUPPORTING HIGH DATA RATES OVER AN ASYNCHRONOUS SERIAL LINE USING DMA, filed August 7, 1997 by Melanie D. Typaldos and Patrick E. Maupin.

U.S. Patent Application. bearing Attorney Docket No. A98201US, entitled UART CHARACTER MATCHING USED FOR ADDRESS MATCHING ON A REGISTER-BY-REGISTER BASIS, filed concurrently by Melanie D. Typaldos.

Turning to Figure 1A, shown is a block diagram of a typical microcontroller M implemented according to the invention. Such a microcontroller is preferably implemented on a single monolithic integrated circuit.

The microcontroller M preferably includes an internal bus 100 coupling, an execution unit 124, system peripherals 174, memory peripherals 176 and serial communication peripherals 172. The execution unit 124 in the disclosed embodiment is compatible with the AM186 instruction set implemented in a variety of microcontrollers from Advanced Micro Devices, Inc., of Sunnyvale, California. A variety of other execution units could be used instead of the execution unit 124. The system peripherals 174 include a watch dog timer (WDT) 104 for generating non-maskable interrupts (NMIs), microcontroller resets, and system resets. An interrupt controller 108 for supporting thirty-six maskable interrupt sources through the use of fifteen channels is also provided as a system peripheral. One disclosed system peripheral is a three channel timer control unit 112. The timer control unit 112 includes three 16-bit programmable timers. Another system peripheral is a general purpose direct memory access (DMA) unit 116 with four channels 0-3. A programmable I/O unit 132 of the microcontroller M supports user programmable input/output signal (PIOs). In the disclosed embodiment, forty-eight PIOs are provided.

The memory peripherals 176 of the disclosed microcontroller include a DRAM controller 170, a glueless interface 168 to a RAM or ROM, and a chip select unit 126. In the disclosed embodiment, the DRAM controller 170 is fully integrated into the microcontroller M. Also in the disclosed embodiment, the chip select unit 126 provides six chip select outputs for use with memory devices and eight chip select outputs for use with peripherals.

A low speed serial port implemented as a universal asynchronous receiver/transmitter (UART) 136 is provided as a serial communication peripheral. The low speed UART 136 is typically compatible with a standard 16550 UART known to the industry. Another serial communication peripheral in the disclosed embodiment is a synchronous serial interface (SSI) 140. Preferably the microcontroller M acts as a master in the synchronous serial interface 140, which is a standard synchronous serial channel.

The microcontroller M in the disclosed embodiment is particularly well suited to communications environments. To this end, the serial communication peripherals 172 of the microcontroller M include a number of high speed communication controllers, including a High-level Data Link Control (HDLC) controller 144, a Universal Serial Bus (USB) controller 146, and a high speed serial port (HSUART) 148. The disclosed HDLC

controller 144 provides four HDLC channels 164. The HDLC channels 164 and the USB controller 146 can be written to and read from by a "SmartDMA" unit 150, a unit which provides for chained buffers that are accessed via pairs of DMA channels. The SmartDMA unit 150 allows for a high degree of packetized transfer without excessive execution unit 124 intervention. The SmartDMA unit 150 preferably consists of four SmartDMA controllers, SmartDMA0-3, that each consists of a pair of DMA channels.

The HSUART 148 serves to form an asynchronous serial link across a bus to devices external to the microcontroller M. The asynchronous nature indicates that the HSUART 148 does not provide a separate clock signal to clock the data. Instead the rate at which data is sent and received must be predetermined or determined through autobauding and independently controlled on sending and receiving ends. This data rate is known as the baud rate. It should be understood that the microcontroller M may include multiple HSUARTs 148.

The disclosed HDLC controller 144 also includes an interface multiplexer 162. This multiplexer 162 couples the four HDLC channels 164, four time slot assignors (TSA) 166, and a number of external buses. Specifically, using the time slot assignors or otherwise, the HDLC channels 164 can be selectively coupled to a pulse code modulation (PCM) highway, a general circuit interface (GCI), an ISDN oriented modular interface revision 2 (IOM-2) serial bus, a data carrier equipment (DCE) serial interface, and other general and specific interfaces that often use packetized communication. Further, the HDLC channels 164 support HDLC, SDLC, Link Access Procedures Balanced (LAPB), Link Access Procedures on the D-channel (LAPD), and PPP, and as noted above, each include an independent time slot assignor 166 for assigning a portion of a serial frame to each HDLC for isochronous or isochronous-type communication.

Turning to Figure 1B, shown are illustrative pinouts for the microcontroller M implemented according to the invention. Illustrated are clock pinouts for the clock 102, address and address/data bus pinouts to the bus interface unit 120, bus status and control pinouts, again generally for the bus interface unit 120, timer control pinouts coupled to the timer control unit 112, USB control and transceiver control pinouts for the USB controller 146, synchronous serial controller pinouts for the synchronous serial interface 140, programmable I/O pinouts for the programmable I/O unit 132, reset control pinouts, memory and peripheral control pinouts coupled to both the chip select unit 126 and the bus interface unit 120, DMA control pinouts for the general purpose DMA unit 116 and the SmartDMA unit 150, HDLC channel/DCE interface/PCM interface pinouts for coupling to the HDLC controller 144, UART pinouts for the low speed UART 136, and high speed UART pinouts for the HSUART 148. All of these pinouts, of course, are illustrative, and a wide variety of other functional units and associated pinouts could be used without detracting from the spirit of the invention. For example, a number of both the communications and general purpose peripherals from Figure 1A could be eliminated, or added to, without detracting from the spirit of the invention.

The techniques and circuitry according to the invention could be applied to a wide variety of microcontrollers and other similar environments. The term "microcontroller" itself has differing definitions in industry. Some companies refer to a processor core with additional features (such as I/O) as a "microprocessor" if it has no onboard memory, and digital signal processors (DSPs) are now used for both special and general purpose controller functions. As here used, the term "microcontroller" covers all of the products, and generally means an execution unit with added functionality all implemented on a single monolithic integrated circuit.

Turning to Figure 2, further illustrated are the DMA unit 116 and the high speed UART 148, here described as an asynchronous serial port (ASP) 148. This is further described in previously incorporated application entitled A METHOD AND APPARATUS FOR SUPPORTING HIGH DATA RATES OVER AN

ASYNCHRONOUS SERIAL LINE USING DMA. The ASP 148 can support an asynchronous serial protocol that employs an "address bit" as an extra serial bit.

Referring to Figure 3, illustrated is an asynchronous serial transmission employing such a protocol. After a start bit 200, the ASP 148 transmits or receives 7 or 8 data bits 202. These are standard data bits sent in an asynchronous protocol, low order bit first. After the 7 or 8 data bits 202, however, the address bit protocol supported by the ASP 148 provides for an optional address bit AB 204. This address bit is typically either the 8th or 9th bit transmitted or received by the ASP 148, and is typically used in applications such as multi-drop applications that require one master device to control multiple slave devices. Although called an address bit, this bit can act as an extended bit that effectively causes interrupts within the slave devices for flow and other control. This address bit protocol is known to the art.

After the optional address bit AB 204, an optional parity bit PB 206 is then followed by a stop bit ST 208, and then an optional second stop bit ST 210.

Returning to Figure 2, the DMA unit 116 and the ASP 148 have additional features according to the invention that may be useful in sending and receiving data that employs an address bit AB 204 as illustrated in Figure 3. These features are best understood by examining a set of registers within the DMA unit 116 and the ASP 148, here shown as DMA registers 212 and ASP registers 214. The DMA registers 212 will be further discussed in conjunction with Figures 6A-6E, and the ASP registers 214 will be further discussed in conjunction with Figures 7A-7G.

BYTE-TO-WORD EXPANSION AND WORD-TO-BYTE COMPRESSION DMA UNIT

In one aspect of the microcontroller M according to the invention, the DMA unit 116 can compress and decompress data during transfers between a source and a destination that have two differing data sizes. For example, the DMA unit 116 can read data byte-by-byte from a source memory block, and write each read byte of data as the low (or high) order byte of consecutive words in a destination memory block. Conversely, the DMA unit can read the low order byte of consecutive words of data from a source block, and then store those read bytes consecutively in a destination block. This is especially helpful in conjunction with the ASP 148 when it is used in an extended read and write mode. As further discussed below in conjunction with Figures 7A-7G, and as previously described in the concurrently pending application METHOD AND APPARATUS FOR SUPPORTING HIGH DATA RATES OVER A ASYNCHRONOUS SERIAL LINE USING DMA, the ASP 148 can read in 7 or 8 bit data units over a serial line, but store that data into a 16-bit word along with both an address bit and status bits corresponding to that I/O read. These are further illustrated in Figure 7G, discussed below, but to summarize, parity errors, overrun errors, framing errors, character matches, breaks, the address bit, and other status can be stored along with 7 or 8 bits of data actually received. When this is performed using DMA, this allows the execution unit 124 to go back and examine the received data to determine where any errors may have occurred or address bits may have been set, and process the data accordingly.

But once errors are examined and address bits are processed, typically the received data no longer needs that associated status. That is, the execution unit 124 can then simply store the data as 7 or 8 bit data within bytes instead of as words.

It is at this point that the DMA unit 116 is especially advantageous. Referring to Figure 4, illustrated is the DMA unit 116 processing data from the ASP unit 148 as discussed. The DMA unit 116 stores this data in a memory 216 as a series of data and status bytes within a block 218, shown as DATA1, STATUS1, DATA2,

STATUS2, etc. So, each 7 or 8 bit data value DATA1-DATAn occupies 16 bits when combined with its associated status byte STATUS1-STATUSn.

The DMA unit 116, however, can then be programmed by the execution unit 124 to perform a transfer from the block 218 to another block 220 in the memory 216, but this block to include simply a number of the consecutive data values DATA1-DATAn but without the associated status. The DMA unit 116 first reads the first data item DATA1 and from the block 218 then writes it to the destination location DATA1 in the block 220. Then, however, the DMA unit 116 increments the destination address within the block 220 by 1 but simultaneously increments the source address in the block 218 by 2. Therefore, the next data item read from the block 218 is DATA2 (STATUS1 is skipped), which is then written as a byte in the destination block 220 immediately after the first data item DATA1. This is repeated, compressing the data such that the original 16-bit data and status values are now simply 8-bit data values.

Turning to Figure 5, the reverse is illustrated for writes by the ASP 148 of external data. Here, a source data block 222 contains data that the microcontroller M should transmit to an external asynchronous serial line. This data includes four data items DATA4 that each occupy 7 or 8 bits within a byte of memory. The DMA unit 116 is programmed to read the first of these data items DATA1 in the block 222, and then write that data item DATA1 into a destination block 224. Then, the source address within the data block 222 is incremented, but the destination address within the destination block 224 is incremented by 2. Thus, the next data item DATA2 is read from the source block 222, but is written 2 bytes later into the destination block 224.

Before that data is actually transmitted, it may be desirable for the execution unit 124 to clear address bits that are stored in the intervening bytes of the data items DATA1-DATA4 in the destination block 224. Here, three address bits 226 are shown cleared, and one address bit 228 set.

The DMA unit 116 is then programmed to perform word-size writes to the ASP 148 using the extended write capability of the ASP 148. The low order bit of the high order byte is then used as the address bit for transmission of data, resulting in four illustrated data items 230, 232, 234, and 236, the first of which has its address bit set and the remainder of which do not.

The DMA unit 116 may include more than one DMA channel, and those DMA channels can be implemented as circular buffers. It would be further possible to implement two channels in Figure 4 such that one channel was reading extended data from the ASP 148 and then writing it into a circular buffer implemented as the buffer 218, and then a second channel was reading from the buffer 218, compressing the data into the destination buffer 220.

By providing compression and decompression of data by the DMA unit 116, the execution unit 124 is relieved of the processing overhead required to strip or add unneeded status information from or to data before the DMA unit 116 sends or receives data to the ASP 148.

Turning to Figures 6A-6E, illustrated are five of the DMA registers 212 employed to implement the compression and decompression according to the invention. In Figure 6A, illustrated is a general purpose DMA channel "x" control register GDxCON0 300. (Note with four channels, the "x" would be from 1 to 4.)

The GDxCON0 register 300 includes a number of standard general purpose DMA control bits, such as a start/stop bit ST 302, an auto start bit AST 304, a terminal count stop bit TC 306, an interrupt enable bit INT 308, a priority field P 310, and a DMA request source field DSEL 312.

The GDxCON0 register 300 also includes a transfer size bit TS 314. This bit selects whether the channel will transfer a byte of data at a time (TS=0, or 1'b0), or will transfer a word at a time, (TS=1'b1). With the

compression and decompression functions according to the invention, the TS bit 314 will generally be set to 0 for byte size transfers.

Turning to Figure 6B, illustrated is a second general purpose DMA control register GDxCON1 316. The GDxCON1 register 316 also includes a number of standard DMA bits, such as a source address space select bit SM/*IO 318, a source address wrap field SAW 320, a destination address space select bit DM/*IO 322, and a destination address wrap field DAW 324. The SM/*IO bit 318 and the DM/*IO bit 322 select whether the source and destination space will be memory or I/O.

When the source or destination address space is to be in memory, generally it is desirable to increment or decrement a source and destination address after the data is sent or received. To this end, two fields are provided, a source increment field SINC 326 and a destination increment field DINC 328. These are 4-bit 2's complements values that are added to the source and destination pointer after each transfer. For example, if the SINC field 326 or the DINC field 328 are 0, there is no increment. If they are equal to a binary 1, or 4'b0001, the fields are incremented by 1. If they equal 4'b0010, the pointers are incremented by 2 bytes, if equal to 4'b0011, incremented by 3 bytes, and so on. For negative values the opposite is true. For example, using 2's complement format, if the SINC or DINC fields 326 or 328 are equal to 4'b1111, the source or destination address will be decremented by 1 after the transfer.

By providing a separate SINC field 326 and DINC field 328, and further by providing that the amount of increment or decrement is independent of the size of the data item, compression or decompression of data using DMA can be affected. This is further discussed below again in conjunction with Figures 4 and 5.

Turning to Figures 6C and 6D, illustrated are source address registers 330 and destination address registers 332. Finally, turning to Figure 6E, illustrated is a transfer count register 334. These are all well understood in conjunction with the DMA design.

Returning to Figures 4 and 5, and given the registers described in Figures 6A-6E, the programming and operation of the DMA unit 116 for compression is better understood. In Figure 4, for the transfer between the ASP 148 and the destination block 218, the DMA unit 116 is programmed with a set of values 336. Specifically, the source is set to be from an I/O location, and to increment by 0 after each transfer (appropriate for an I/O location), and the destination is set to memory and is set to increment by 2 after each transfer. Further, the data size is set to a word. This provides that each transfer will be a 16-bit extended serial read value from the ASP 148, which is stored word-by-word in the destination block 218.

For the compression function, the DMA unit 116 is programmed with a set of values 338. Specifically, the source is set in memory (and the source address 330 will be set to the start of the source block 218), and is set to increment by 2 after each transfer. The destination is also set in memory, but the destination address is incremented by only 1 after each transfer. Finally, the transfer size is set for 1 byte. This means that 1 byte will be read from the source block 218 and written to the destination block 220 as DATA1. The source address 330, however, will then be incremented by 2, while the destination address 332 will only be incremented by 1. Then a second byte of data, DATA2, will be read from the source block 218 and written immediately after DATA1 in the destination block 220. Thus, unneeded status information in STATUS1 is skipped.

Of note, the destination block 220 could be programmed to actually overlap with the source block 218 because after the data is read from the source block 218, it is no longer needed. Thus, the source buffer can literally be used as the destination buffer, saving memory space.

Turning to Figure 5, the expansion capability of the DMA unit 116 is illustrated by a set of programmed values 340. Here, the source block 222 of 8-bit data is expanded by setting the DMA unit 116 to transfer from memory to memory, by setting the SINC field 326 to a value of 1, but setting the DINC field 328 to a value of 2. The transfer size bit TS 314 is set to 0, such that data is transferred a byte at a time. With this programming, a data byte is read from the source block 222 and written to the destination block 224, but the next data byte consecutively read from the source block 222 is written 2 bytes later in the destination block 224. The execution unit 124 can then appropriately set or reset the address bits AB, in the destination block 224, and institute a second DMA transfer to the ASP 148.

Here, a set of programmed values 342 is used, which provide for a source in memory and destination in I/O space, and provide for a source increment SINC 326 value of 2 and no destination increment. Further, the transfer size bit TS 314 is set to 1 for word transfers. This results in the extended data being written to the ASP 148, so that it can control the 9th, address bit on its transmission of serial data.

Therefore, by providing independent controllability of the source increment and destination increment independent of the size of the data being transferred, compression and decompression of memory can be affected. This could be used for a variety of other functions in which it was desirable to compress word data into byte data, or vice versa.

CHARACTER MATCHING USING ADDRESS BITS

Turning to Figures 7A-7G, illustrated are certain of the ASP registers 214 that implement another feature according to the invention. Figure 7A illustrates 6 character match bytes employed by the ASP 148 for matching characters of received data. Specifically, three registers HSPM0 400, HSPM1 402, and HSPM2 404 contain 6 character match values MCHR0 406 through MCHR5 415.

The operation of these byte size values MCHR0 406 through MCHR5 415 is better understood in conjunction with Figure 7G, which illustrates an asynchronous serial port 418 receive register HSPRXD 416. This register contains a 7 or 8 bit value received data RDATA 418, but also contains 8 status bits 420 through 434.

Historically, only the received data RDATA 418 was compared to the match character values MCHR0 406 through MCHR5 415. When the RDATA value 418 matched one of those matched character values, a status bit was set in a status register, illustrated below in conjunction with Figure 7D, and an interrupt was generated if an appropriate mask value was set, illustrated below in conjunction with Figure 7E. Historically, the address bit AB 204, would be reflected as first a bit within the extended data of the HSPRXD register 416 (here the address bit value AB 434), and could also be found in the status register discussed below in conjunction with Figure 7D. The address bit would not, however, be used to match with one of the character matched values MCHR0 406 through MCHR5 415.

According to the invention, additional bits are provided such that when set, the address bit value AB 434 is matched in addition to the RDATA value RDATA 418.

This is better understood in conjunction with a high speed serial port control register HP CON0 436, illustrated in Figure 7C. This register in part provides configuration settings to determine whether an address bit is going to be employed (indicated by an ABEN bit 442), whether the parity bits PB 206 will be used (indicated by a parity enable bit PEN 440), whether 7 or 8 bit data will be used and thus whether the D7 bit of the bits 202 will be used (indicated by enable bit D7 444), and whether the second stop bit ST2 210 will be used (indicated by a second

stop bit enable bit STP2 446). When data writes, an address bit set or reset bit AB 438 sets or resets the transmitted address bit, as long address bit communication is enabled (controlled by the ABEN bit 442).

All of these bits 438 through 446 affect the length of a transmitted frame. In the disclosed embodiment, and historically, however, even if the receive data 418 is 7-bit data (the D7 bit 444 false) and an address bit is enabled (the ABEN bit 442 is true), the RDATA 418 would not include the address bit. Instead, that would still be stored in the high order byte of the HSPRXD register 416 as the address bit AB 434. Thus, that bit would never be matched with any of the match characters MCHR0 406 through MCHR5 415.

According to the invention, however, additional match character address bit MATCH bits are provided in a second control register HSPCON1 448. Specifically, this register 448 includes a match enable bit MEN 450, which when true, enables the character matching of the MCHR0 406 through MCHR5 415. According to the invention, however, additional character matching bits are provided. Three match address bits MAB2 452, MAB1 454, and MAB0 456 effectively become an address bit matching bit for the three match registers HSPM0 400 through HSPM2 404 when the address bit is enabled by the ABEN bit 442 of the control register HSPCON0 436.

For example, when the MAB2 bit 452 is 0 and the ABEN bit 442 is 1, a received character must have its address bit AB 204 clear to match characters in the HSPM2 register 404. When the MAB2 bit 452 is set, and the ABEN bit 442 is true, a received character must have its address bit AB 204 set in order to match characters in the HSPM2 register 404. The MAB1 bit 454 and the MAB0 bit 456 similarly correspond to the HSPM1 register 402 and the HSPM0 register 400.

Thus, these additional bits in the HSPCON1 register 448 provide for address matching for corresponding ones of the HSPM0 register 400 through HSPM2 register 404. The ASP 148 could be set up where one of the registers HSPM0 400 through HSPM2 404 was set for character matching, while the other two were set for address matching.

The HSPCON1 register 448 further includes an extended write bit 458 and an extended read bit 460, which provide, when true, that 16-bit data will be read from a transmit register HSPTXD 462 (see Fig. 7F) or received in the receive register HSPRXD 416 (see Fig. 7G). When these values are disabled, only 8-bit data is stored in those registers, and the address bit and the match status can only be determined by reading a status register HPSTAT 464 (see Fig. 7D). That status register 464, among other things, includes a match bit MATCH 466 which is set true if the incoming data is a character or address match (as appropriate) to one of the HSPM0-HSPM2 registers 400-402 and an address bit AB 468 which is set true if the incoming data has its address bit set. An interrupt mask register HSPIMSK 470 correspondingly passes on the match bit MATCH 466 and the address AB 468 as interrupts.

Turning to Figure 8, illustrated is a multidrop system in which the address bit matching according to the invention could be used. A host system 500 transmits serial data to multiple multidrop peripherals 502, 504, and 506. Each of these peripherals includes a microcontroller M of its own 508, 510, and 512. Each of these can be programmed such that it had a unique address 1, 2, and 3 for the peripherals 502, 504, and 506. Each can be placed into an idle mode where received characters are effectively dumped by each of the peripherals 502, 504, and 506 until one of the peripherals has an address match. For example, the microcontroller 508 of the peripheral 502 could be programmed such that the MCHR0 byte 406 contained the appropriate addressing information, including the "1" address assigned to the peripheral 502, and its associated address matching bit MAB0 456 would be set, along with the address bit enable ABEN 442. In this way, when the host 500 sent a frame of asynchronous data with the appropriate address matching MCHR0 406 to the peripheral 502, that data would match the character in

MCHR0 406 and the address bit being set would match the MAB0 456 bit. Therefore, the match bit MATCH 466 would be set in the HPSTAT register 464, and, assuming the match interrupt was set to pass in the HSPIMSK register 470, an interrupt would be passed to the execution unit 124 so that the peripheral 502 could be awaken and set to process subsequent data.

Thus, providing both character matching and address matching independently for certain of the match address registers, the asynchronous serial port according to the invention could not only detect the start and end of data frames based on matchable character data, but can also detect when a particular microcontroller is being addressed in a multidrop system by also matching the address bit. This provides for added flexibility in applications of the microcontroller M.

While techniques according to the invention have been described in conjunction with asynchronous communication ports, they can similarly be used with synchronous communication ports for compression and decompression and for address matching.

The foregoing disclosure and description of the invention are illustrative and explanatory thereof, and various changes in the details of the illustrated apparatus and construction and method of operation may be made without departing from the spirit of the invention.

CLAIMS:

We claim:

1. A microcontroller including an execution unit (124) for executing instructions, the microcontroller characterized by:

a direct memory access (DMA) controller (116) that includes a source address register, a destination address register, a source increment field, and a destination increment field,

wherein, the source increment field and the destination increment field independently control the incrementing of the source address register and the destination address register, and

wherein the incrementing of the source address register and the destination address register are further independent of transfer size such that data can be compressed or expanded during DMA transfers.

2. The microcontroller of claim 1, further comprising:

a serial port (148) for receiving serial data, the serial port coupled to the DMA controller (116),

wherein the DMA controller (116) is configurable to transfer both data and status from the serial port (148) to memory, and

wherein the DMA controller (116) is configurable to perform a subsequent DMA from memory to memory where the status portion of the received data is discarded.

3. The microcontroller of claim 2, wherein the serial port (148) is an asynchronous serial port.

4. The microcontroller of claim 2, wherein the status information received by the serial port (148) further comprises address bit status information.

5. The microcontroller of claim 4, wherein the serial port (148) is a synchronous serial port.

6. The microcontroller of claim 1, wherein the DMA controller (116) is configurable to increment the source address by two and increment the destination address by one for each transfer of a byte of data from the source address to the destination address.

7. A direct memory access (DMA) system, characterized by:

a direct memory access (DMA) controller (116) that includes a source address register, a destination address register, a source increment field, and a destination increment field,

wherein, the source increment field and the destination increment field independently control the incrementing of the source address register and the destination address register, and

wherein the incrementing of the source address register and the destination address register are further independent of transfer size such that data can be compressed or expanded during DMA transfers.

8. The DMA system of claim 7, further comprising:

an serial port (148) for receiving serial data, the serial port coupled to the DMA controller (116),

wherein the DMA controller (116) is configurable to transfer both data and status from the serial port to memory, and

wherein the DMA controller (116) is configurable to perform a subsequent DMA from memory to memory where the status portion of the received data is discarded.

9. The DMA system of claim 8, wherein the serial port (148) is an asynchronous serial port.

10. The DMA system of claim 8, wherein the status information received by the asynchronous serial port further comprises address bit status information.

11. The DMA system of claim 10, wherein the serial port (148) is a synchronous serial port.

12. The DMA system of claim 7, wherein the DMA controller (116) is configurable to increment the source address by two and increment the destination address by one for each transfer of a byte of data from the source address to the destination address.

13. A DMA controller including a same address register and a destination address register, the DMA controller (116) characterized by:

a means for transferring data from the source address to the destination address and then modifying the source address and destination address such that a portion of the data located in a block pointed to by the source address is stripped during transfer from the source address to the destination address.

14. A DMA controller (116) including a source address register and a destination address register, the DMA controller (116) characterized by:

a means for transferring data from the source address to the destination address and then modifying the source address and destination address such that a portion of the data located in a block pointed to by the destination address is skipped during transfer from the source address to the destination address.

15. A method for compressing data within memory employing a DMA controller (116) comprising the steps of:

setting the DMA controller (116) to point to a source address block and to a destination address block;

setting the DMA controller (116) to increment a source address by two bytes and a destination address by one byte for each transfer from the source address block to the destination address block;

setting the DMA controller (116) to transfer one byte of information per transfer; and

performing the DMA, such that every other byte of information from the source address block is discarded upon being written into the destination address block.

16. The method of claim 15, wherein the source address is set to the same location as the destination address.

17. A method for compressing received extended serial data, comprising the steps of:

receiving serial data and associated status through an asynchronous serial port (148);

transferring the serial data and associated status to a memory location;

repeating the receiving and transferring steps for a plurality of data and status to a plurality of memory locations such that both data and status are stored in the plurality of memory locations; and

performing a direct memory access transfer data from the plurality of memory locations which includes discarding the status associated with the data and storing the data without the status in a destination memory location.

18. The method of claim 17, wherein the status is stored in 8 bits and the data is stored in 8 bits, and the DMA transfer strips the status from the data by transferring one byte of data from the source to the destination, incrementing a source address pointer by 2 and a destination address pointer by 1, and repeating the transfer step.

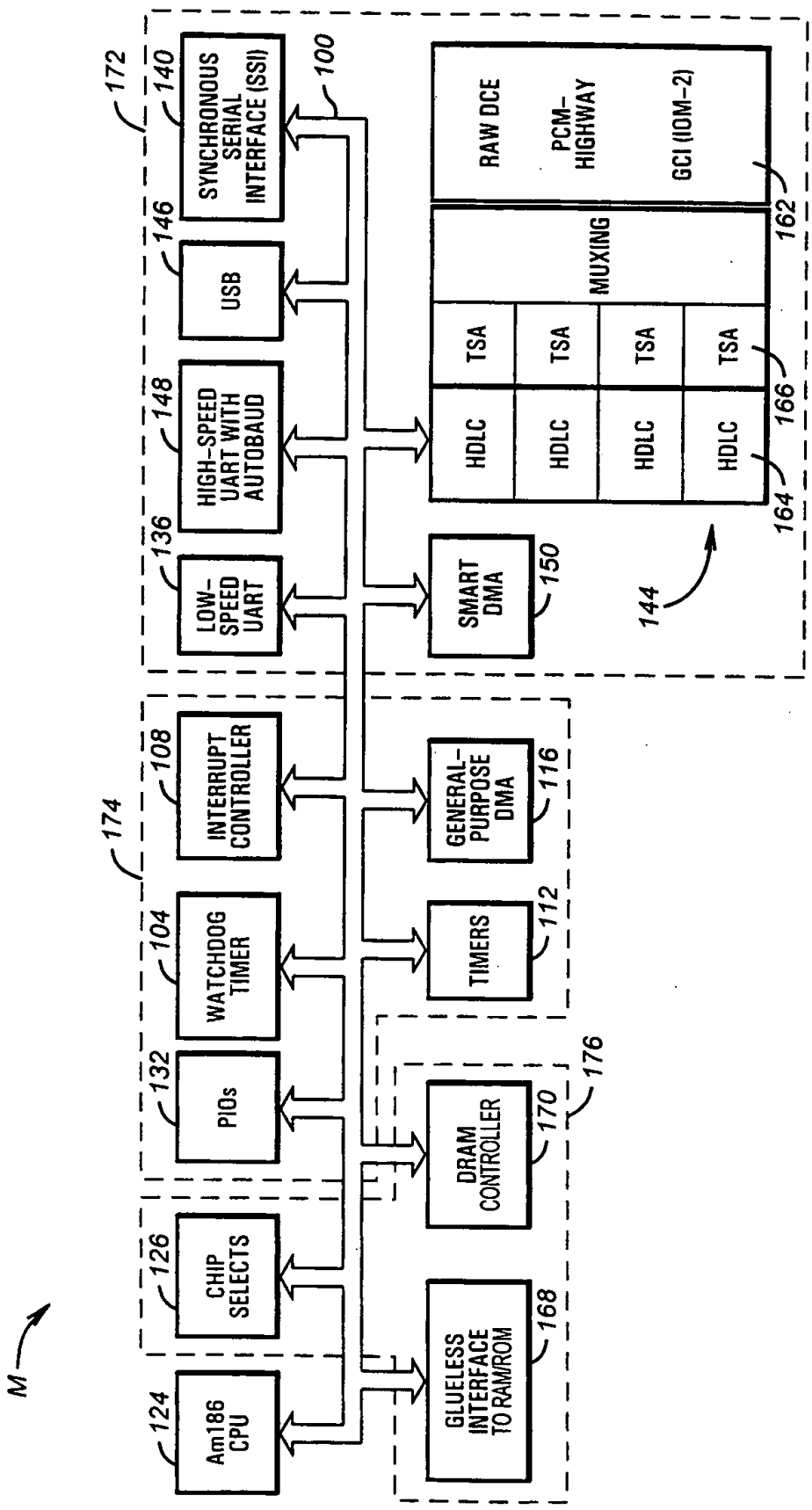

M
172
174
176
144
Am186 CPU
124
CHIP SELECTS
126
PIOs
132
WATCHDOG TIMER
104
INTERRUPT CONTROLLER
108
LOW-SPEED UART
136
HIGH-SPEED UART WITH AUTOBAUD
148
USB
146
SYNCHRONOUS SERIAL INTERFACE (SSI)
140
100
GLUELESS INTERFACE TO RAM/ROM
168
DRAM CONTROLLER
170
TIMERS
112
GENERAL-PURPOSE DMA
116
SMART DMA
150
HDLC
HDLC
HDLC
HDLC
164
166
TSA
TSA
TSA
TSA
MUXING
162
RAW DCE
PCM-HIGHWAY
GCI (IOM-2)

FIG. 1A

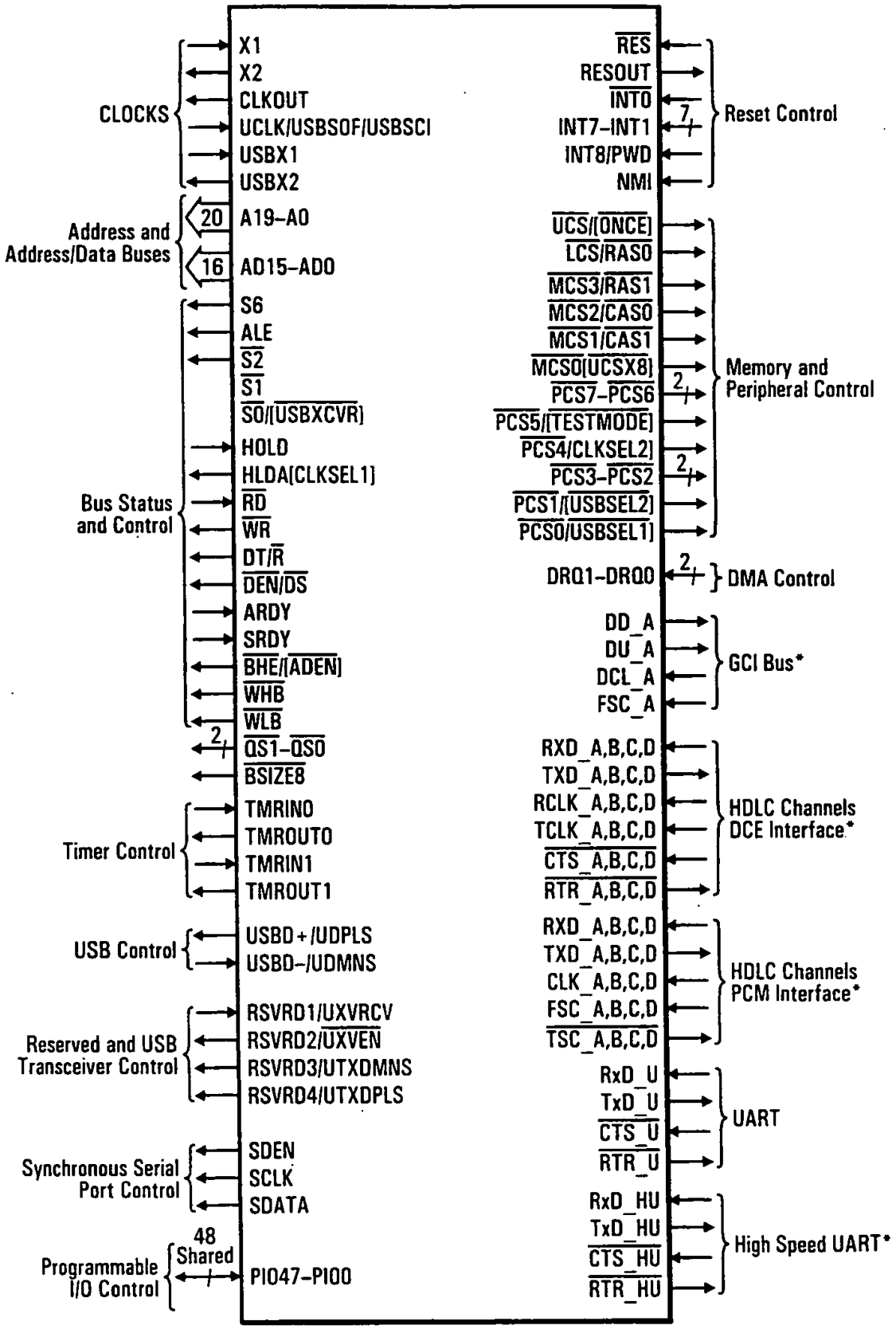

CLOCKS
X1
X2
CLKOUT
UCLK/USBSOF/USBSCI
USBX1
USBX2
Address and Address/Data Buses
20
A19–A0
16
AD15–AD0
Bus Status and Control
S6
ALE
S2
S1
S0/[USBXCVR]
HOLD
HLDA[CLKSEL1]
RD
WR
DT/R
DEN/DS
ARDY
SRDY
BHE/[ADEN]
WHB
WLB
2
QS1–QS0
BSIZE8
Timer Control
TMRIN0
TMROUT0
TMRIN1
TMROUT1
USB Control
USBD+/UDPLS
USBD–/UDMNS
Reserved and USB Transceiver Control
RSVRD1/UXVRCV
RSVRD2/UXVEN
RSVRD3/UTXDMNS
RSVRD4/UTXDPLS
Synchronous Serial Port Control
SDEN
SCLK
SDATA
Programmable I/O Control
48 Shared
PIO47–PIO0
RES
RESOUT
INT0
7
INT7–INT1
INT8/PWD
NMI
Reset Control
UCS/[ONCE]
LCS/RAS0
MCS3/RAS1
MCS2/CAS0
MCS1/CAS1
MCS0/UCSX8]
2
PCS7–PCS6
PCS5/[TESTMODE]
PCS4/CLKSEL2]
2
PCS3–PCS2
PCS1/[USBSEL2]
PCS0/USBSEL1]
Memory and Peripheral Control
2
DRQ1–DRQ0
DMA Control
DD_A
DU_A
DCL_A
FSC_A
GCI Bus*
RXD_A,B,C,D
TXD_A,B,C,D
RCLK_A,B,C,D
TCLK_A,B,C,D
CTS_A,B,C,D
RTR_A,B,C,D
HDLC Channels DCE Interface*
RXD_A,B,C,D
TXD_A,B,C,D
CLK_A,B,C,D
FSC_A,B,C,D
TSC_A,B,C,D
HDLC Channels PCM Interface*
RxD_U
TxD_U
CTS_U
RTR_U
UART
RxD_HU
TxD_HU
CTS_HU
RTR_HU
High Speed UART*

FIG. 1B

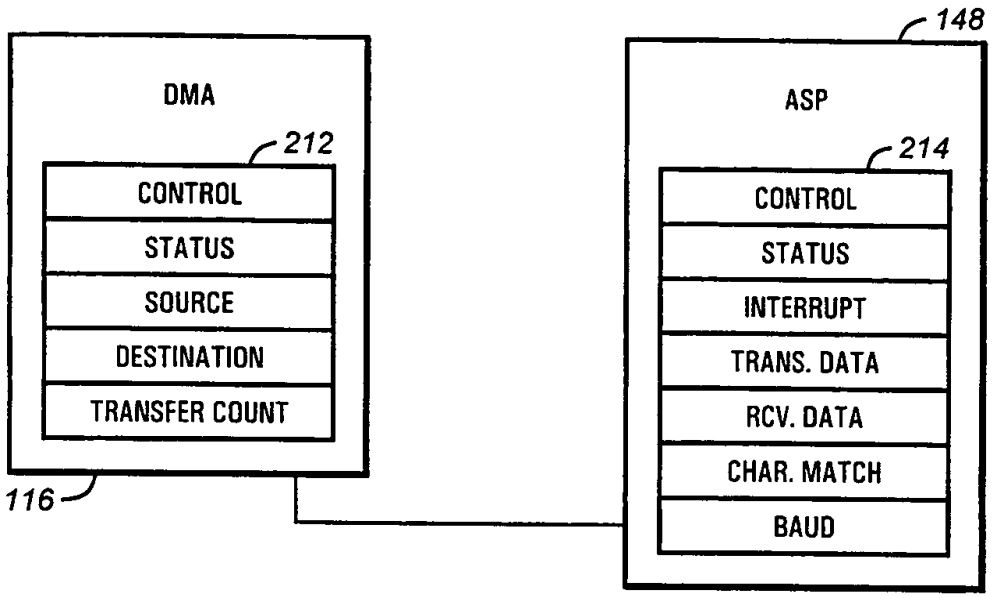

148
DMA
212
CONTROL
STATUS
SOURCE
DESTINATION
TRANSFER COUNT
116
ASP
214
CONTROL
STATUS
INTERRUPT
TRANS. DATA
RCV. DATA
CHAR. MATCH
BAUD

FIG. 2

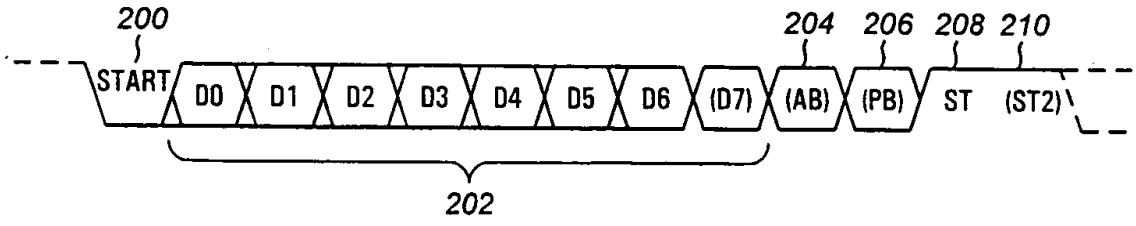

200
204
206
208
210
START
D0
D1
D2
D3
D4
D5
D6
(D7)
(AB)
(PB)
ST
(ST2)
202

FIG. 3

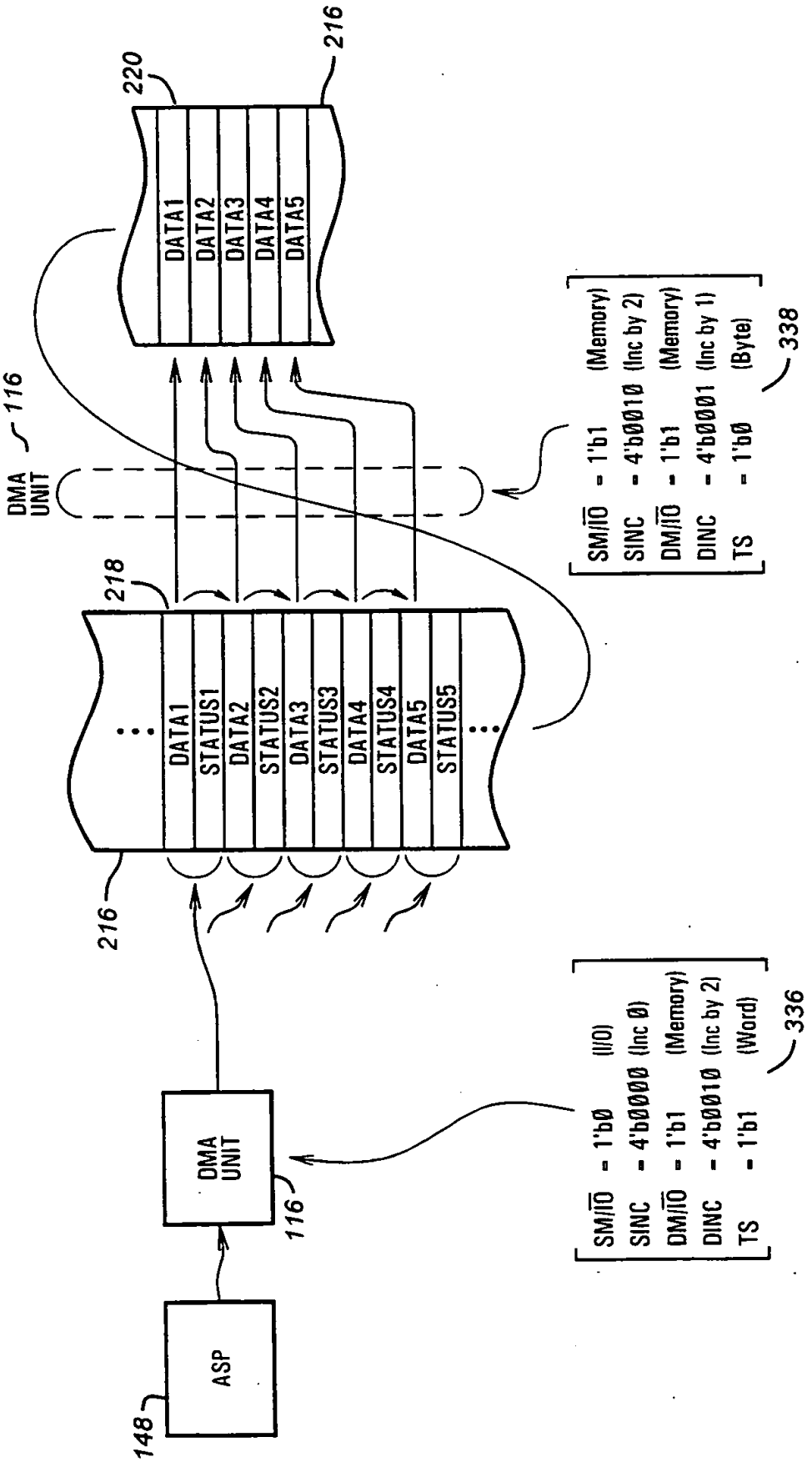

ASP
148
DMA UNIT
116
DMA UNIT
116
216
218
DATA1
STATUS1
DATA2
STATUS2
DATA3
STATUS3
DATA4
STATUS4
DATA5
STATUS5
216
220
DATA1
DATA2
DATA3
DATA4
DATA5
SM/IO = 1'b0 (I/O)
SINC = 4'b0000 (Inc 0)
DM/IO = 1'b1 (Memory)
DINC = 4'b0010 (Inc by 2)
TS = 1'b1 (Word)
336
SM/IO = 1'b1 (Memory)
SINC = 4'b0010 (Inc by 2)
DM/IO = 1'b1 (Memory)
DINC = 4'b0001 (Inc by 1)
TS = 1'b0 (Byte)
338

FIG. 4

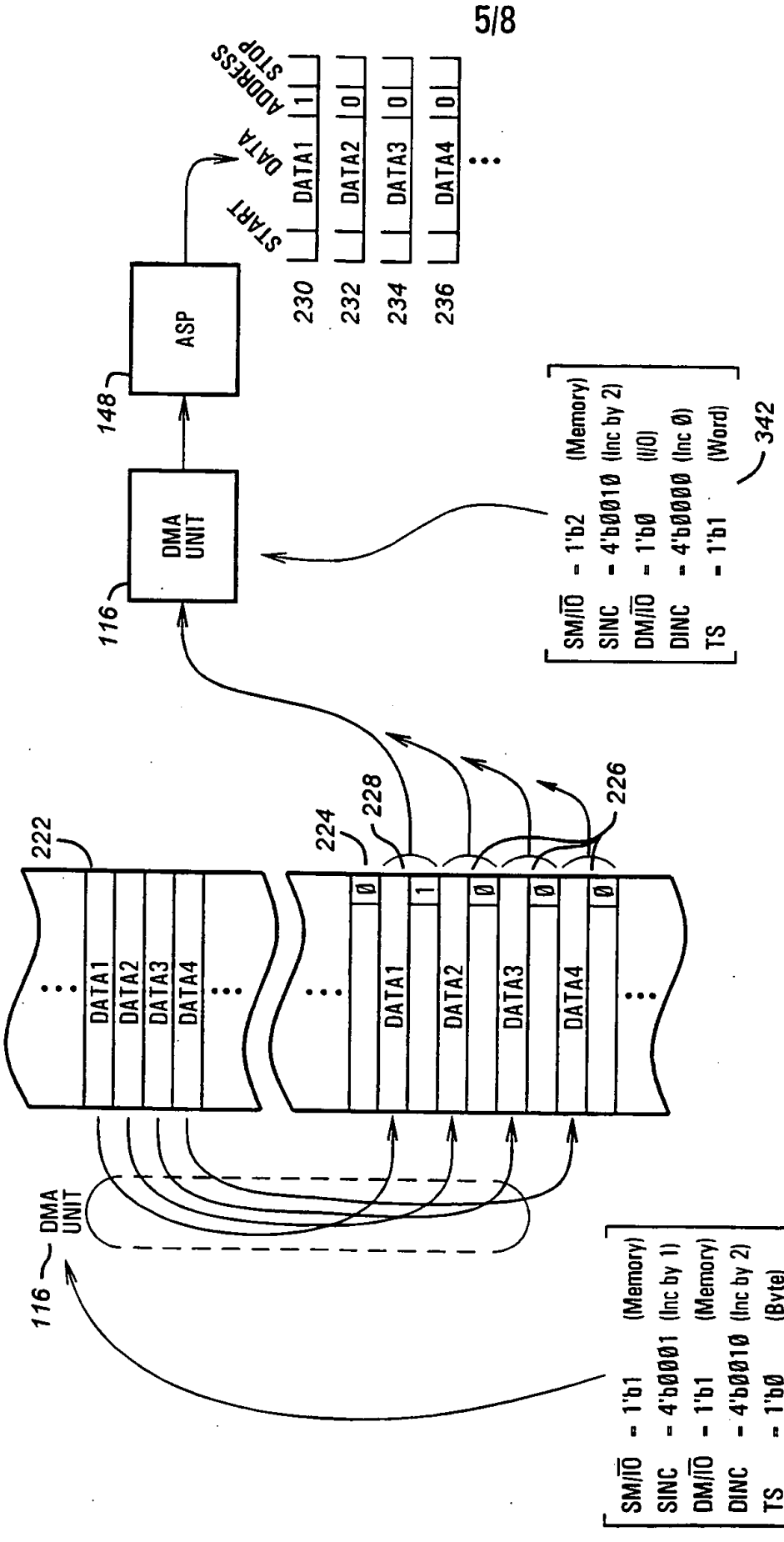

START
DATA
ADDRESS
STOP
DATA1
DATA2
DATA3
DATA4
1
0
0
0
230
232
234
236
ASP
148
DMA UNIT
116
SM/IO = 1'b2 (Memory)
SINC = 4'b0010 (Inc by 2)
DM/IO = 1'b0 (I/O)
DINC = 4'b0000 (Inc 0)
TS = 1'b1 (Word)
342
222
224
228
226
0
1
0
0
0
SM/IO = 1'b1 (Memory)
SINC = 4'b0001 (Inc by 1)
DM/IO = 1'b1 (Memory)
DINC = 4'b0010 (Inc by 2)
TS = 1'b0 (Byte)
340

FIG. 5

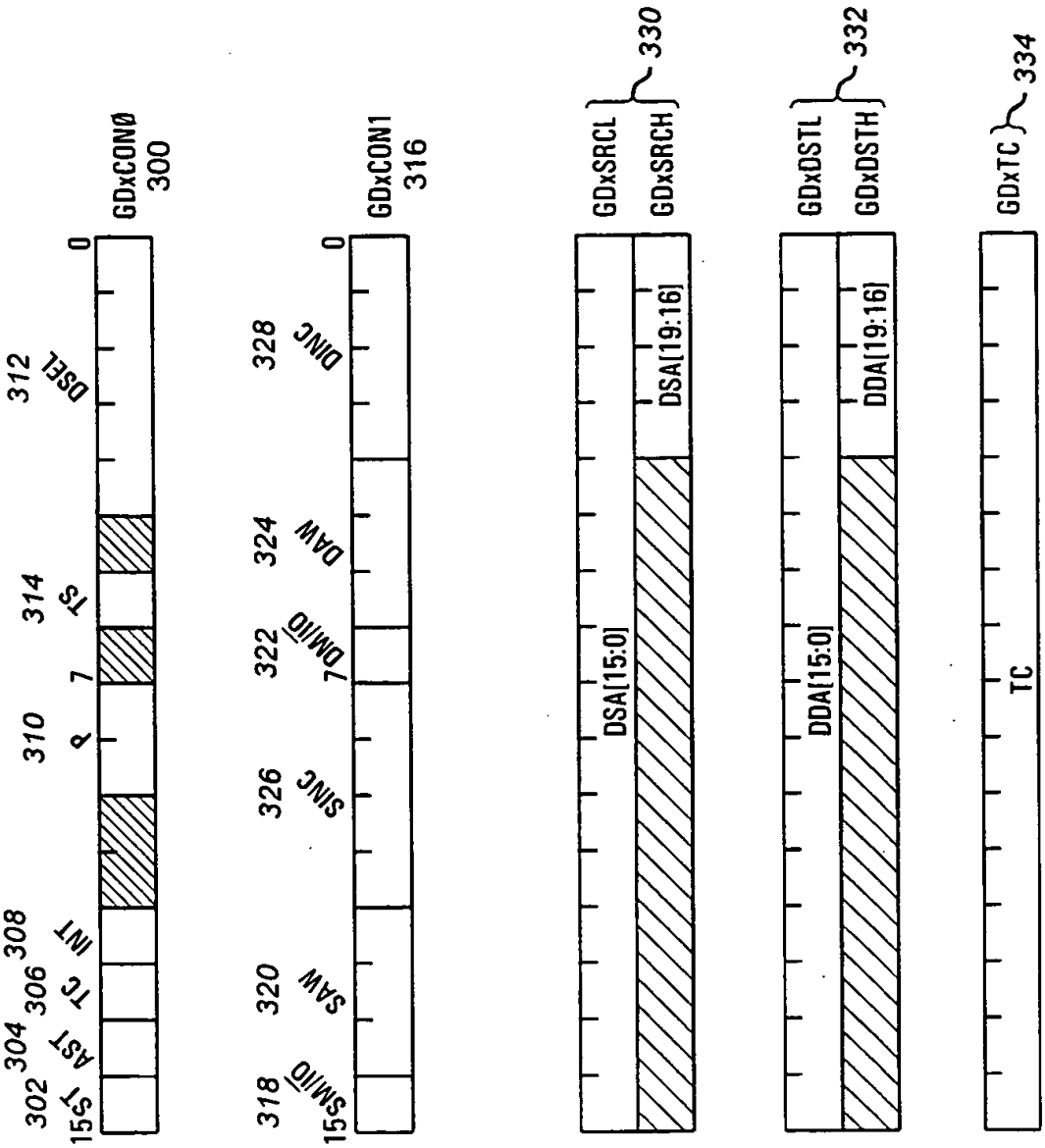


FIG. 6A

FIG. 6B

FIG. 6C

FIG. 6D

FIG. 6E

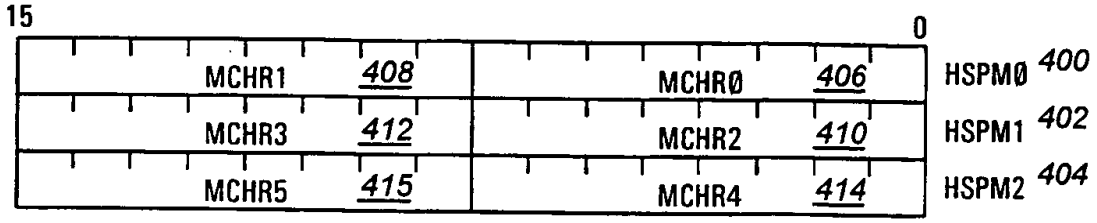


FIG. 7A

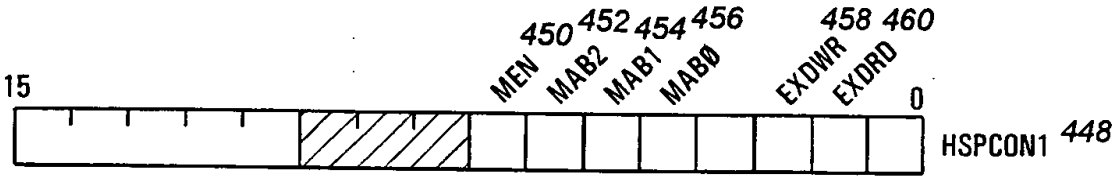


FIG. 7B

FIG. 7C

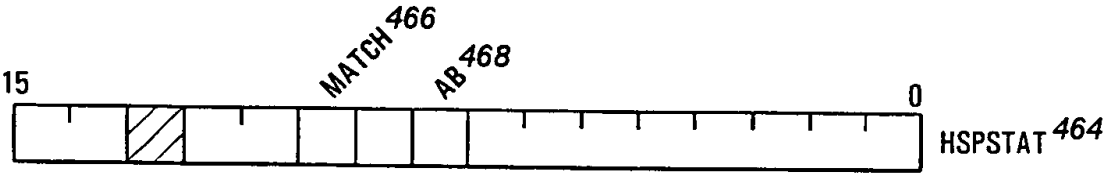


FIG. 7D

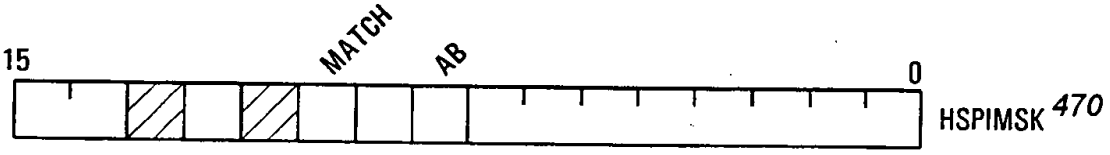


FIG. 7E

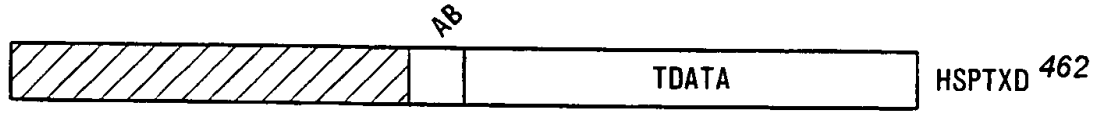


FIG. 7F

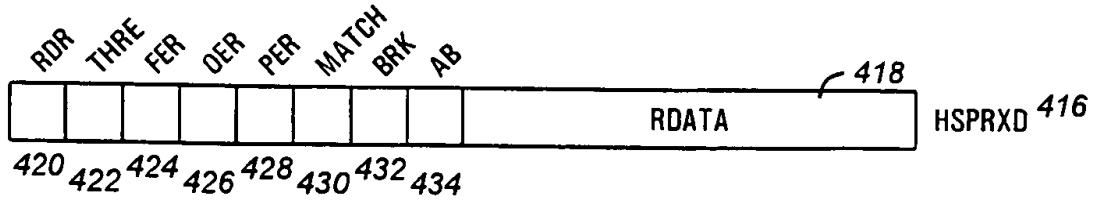


FIG. 7G

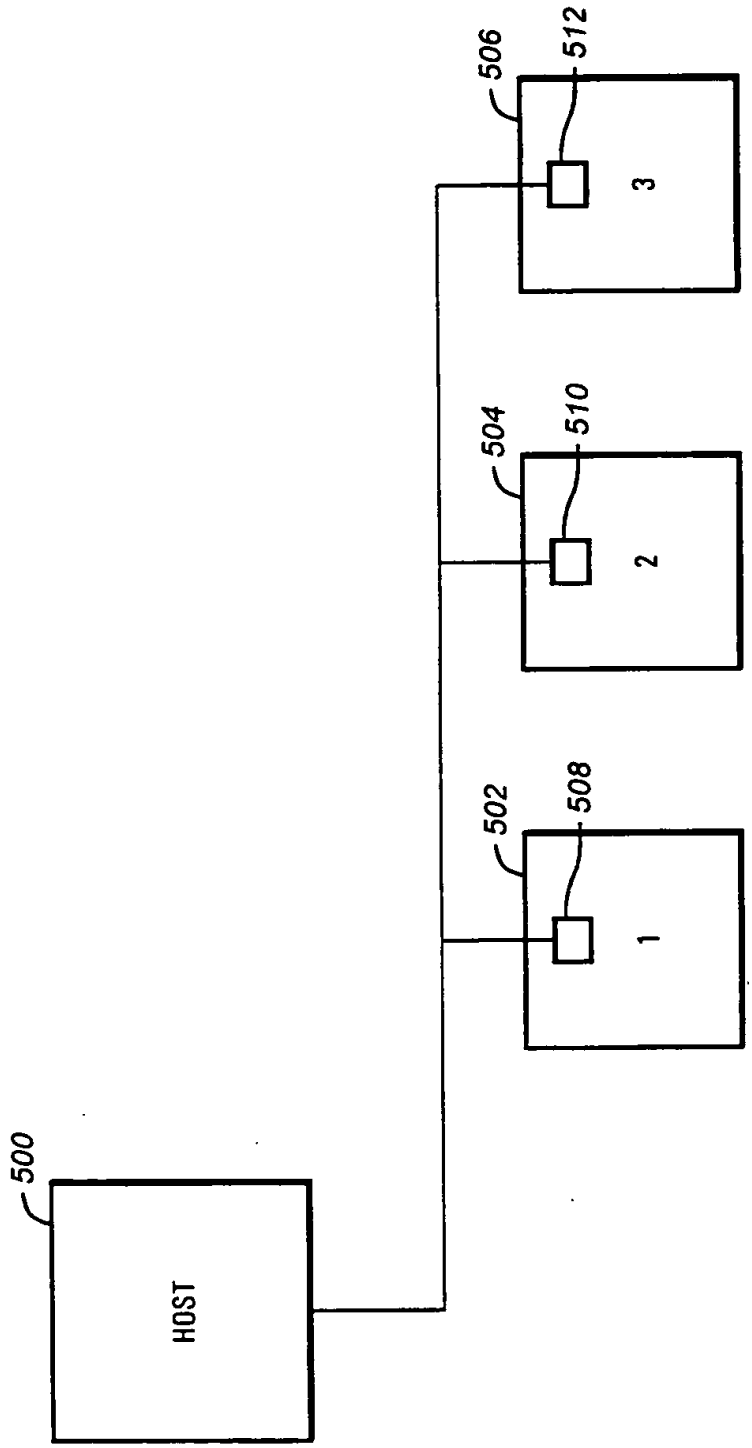


FIG. 8

INTERNATIONAL SEARCH REPORT

Inr tional Application No
PCT/US 99/04610

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F13/28

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 98 11729 A (SILICON GRAPHICS INC) 19 March 1998 (1998-03-19) page 5, line 14 - page 6, line 18 page 8, line 20 - page 9, line 3 page 10, line 37 - page 11, line 32; claim 5	1,7,15
X	US 5 737 638 A (DELP GARY S ET AL) 7 April 1998 (1998-04-07) column 4, line 31 - line 57 column 7, line 35 - line 57	1,7

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
29 July 1999	06/08/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Nygren, P

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

In tional Application No
PCT/US 99/04610

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9811729 A	19-03-1998	US 5768445 A	16-06-1998
		AU 3602797 A	02-04-1998
		EP 0925687 A	30-06-1999
US 5737638 A	07-04-1998	NONE	

Form PCT/ISA/210 (patent family annex) (July 1992)